SECTION 4
DESIGN DETAILS

4.1 DESIGN METHOD. The Booch method is a software-development method used to develop and communicate the design of a system that will be implemented primarily in software. A software-development method is a standardized means of presenting and communicating the requirements of a system and the design decisions. The Booch method is an object-oriented method based on proven heuristics for developing quality software that not only provides an effective design but also supports that design and the development of future systems. Refer to Appendix B for definitions of notations used in the Booch Method.

In an object-oriented method the basic unit of design is the object. This allows software to maintain a one-to-one mapping with the real world making the system easier to understand and maintain. Object-oriented design provides a model to support solid analysis and design, and it allows the developers and implementors to enhance, correct, and maintain the same consistent model from the beginning of analysis through coding and implementation.

The Booch method consists of three steps:

1. **Requirements Analysis.**
   Objective: Provides the basic charter for the system's functions.
   Deliverables:
   a. Problem statement, which describes the system.
   b. System charter, which outlines the responsibilities of the system.
   c. System function statement, which outlines the key use cases of the system.

2. **Domain Analysis.**
   Objective: Provides the key logical structure of the system. Identifies all major objects in the domain, including all data and major operations that will be needed to carry out the system functions.
   Deliverables:
   a. Class diagrams, which identify key classes, or types, of the domain.
   b. Class diagram with relationships.
   c. Class diagram with operations.
   d. Class diagram with attributes.
   e. Class specifications, which contain all semantic definitions of the classes, their relationships, their attributes, and their key system functions.
   f. Object-scenario diagrams, which illustrate how the objects will interact to carry out key system functions.

3. **System Design.**
   Objective: Provides the key physical structure of the system, maps the logical structure to it, and leads to working executable releases.
   Deliverables:
   a. Architectural descriptions, which capture major design decisions.

b. Class-category diagrams, which model the partioning of the system into high-level groupings of classes and objects.
c. Design class diagrams, which show the abstractions of the physical implementation, detailed data types and structures, and the mapping of the logical abstractions to the physical abstractions.

The analysis focuses on understanding the domain while the design focuses on how the domain requirements can be implemented. The domain analysis (presented in this *System Specification for the DSRS*) is expanded into the system design (presented in the *Maintenance Manual for the DSRS*) which is a working implementation. The domain analysis is reusable across more than one implementation, such as for multiple platforms and different Graphical User Interfaces (GUIs).

4.2 REQUIREMENTS ANALYSIS. Requirements Analysis is the process of determining what the system should do. It is a high-level stage that identifies the key functions the system is to perform, defines the scope of the domain that the system will support, and documents the key practices and policies of the enterprise that the system must support. During this stage, the problem statement, the system charter, and the system function statement for DSRS are defined.

4.2.1 Problem Statement. The system to be modeled is a Reusable Asset (RA) repository. The goal is to facilitate the reuse of assets using an automated mechanism for obtaining RAs. The DSRS is an RA repository that may be implemented at multiple sites, and these sites may be interoperable.

To facilitate the reuse of assets, the DSRS provides the capability to search for RAs. The system provides a catalog of available RAs and mechanisms to aid the user in locating an RA.

The user connects to a DSRS server to access the current RA information and RA files based on user privileges. When not connected to a DSRS server, the user is able to access RA information from a local database. A user specifies search criteria for RA IDs in the RA catalog to generate a candidate RA list. The user can search by RA IDs, RA Names, keyword, or viewing an indexed list of RAs on the World Wide Web (WWW). The user has the capability to save a candidate RA list and restore a previously saved candidate RA list.

The user can browse RA information for any of the RAs on the candidate list. The user can view RA information such as a list of RA Files, RA Metrics, list of Related RAs, and information displayed on the RA WWW Catalog Page.

The user can extract RA Files for any of the RAs on the candidate list. RA files can be any electronic file, including an abstract, source, documents, and problem reports.

The DSRS provides the capability for the Supervisor/Librarian to maintain catalog information and configurable RA attributes, control access to the repository, and review tracking data.

The Supervisor/Librarian maintains RA information in the catalog, including the RA Domain Assignments, list of RA Files, RA Metrics and list of Related RAs. RA information can be shared with other DSRS sites to increase the availability of assets.

The Supervisor/Librarian maintains the configurable RA attributes used to establish RA information. These configurable attributes include the domains and the types of relationships and metrics.

The Supervisor/Librarian controls access to the DSRS server by defining user information. The Supervisor/Librarian controls access to the RAs and RA information by defining Sites and Groups and establishing assignments among Groups, Sites, Users and RAs.

The Supervisor/Librarian tracks entries in logs and generates custom or predefined reports on the repository data.

4.2.2 System Charter Statement. A System Charter is a concise summary of the major responsibilities of a system.

The DSRS will be able to:

- Maintain the RA information stored in the catalog and the information that supports the operation of the system.

- Provide session capability for a user to search and browse RA metadata in the catalog and extract RAs from an on-line repository.

4.2.3 System Function Statement. The System Function Statement is a summary of the user cases of the system. Each use case describes some piece of functionality or some action that the system must support. The following paragraphs describe the system function statements for the DSRS User Tool, the DSRS Librarian Tool, and the DSRS Server:

4.2.3.1 DSRS User Tool System Function Statement:

- Select a search on the catalog by List RA IDs, List RA Names, Find Keywords, or Catalog Indexes on the World Wide Web (WWW) to build a list of candidate RAs.

- Select a RA entry from the list of candidates to browse the catalog metadata by File List, Metrics, Related RA List, or view the catalog page on the WWW.

- Select RA entries from the list of candidates to graphically analyze metrics.

- Select RA entries from the list of candidates to extract RA Files

4.2.3.2 DSRS Librarian Tool System Function Statement:

- Maintain Repository Framework to include the domains, types of relationships, and types of metrics.

- Write an import file containing RA metadata to the catalog.

- Create an export file containing RA metadata that can be imported into another DSRS catalog.

- Maintain group information and group assignments among sites, users, and RAs.

- Maintain user information.

- Maintain the entries in the Usage Log.

- Maintain site information for the local site, remote DSRS sites and foreign sites.

- Provide predefined and customized report capability on data in the repository.

4.2.3.3 DSRS Server System Function Statement:

- Provide connectivity capability to a remote or foreign DSRS server site to gain access to the information stored in the catalog.

- Manage file transfer requests from the DSRS User Tool, remote DSRS sites, and foreign sites.

- Process catalog requests for RA metadata to include RA Metrics, list of related RAs, list of RA Files, and other RA Information in the catalog.

4.3 DOMAIN ANALYSIS. Domain Analysis is the process of defining a precise, concise, and object-oriented model of that part of the real-world enterprise that is relevant to the system (the problem domain). It is through this process that the developers gain the detailed knowledge of the problem domain needed to create a system capable of carrying out the required functions. The products of this process are: key classes in the DSRS, DSRS class diagram with relationships, DSRS class diagram with operations, DSRS class diagram with attributes, DSRS object-scenario diagrams, and DSRS class specifications.

4.3.1 Key Classes in the DSRS. Figure 4-1 shows the key classes in the DSRS which are the major domain items of interest. A class is denoted by a cloud icon and a word or phrase that describes a unique and distinct name. Section 4.3.5 describes the definitions and specifications of each class.

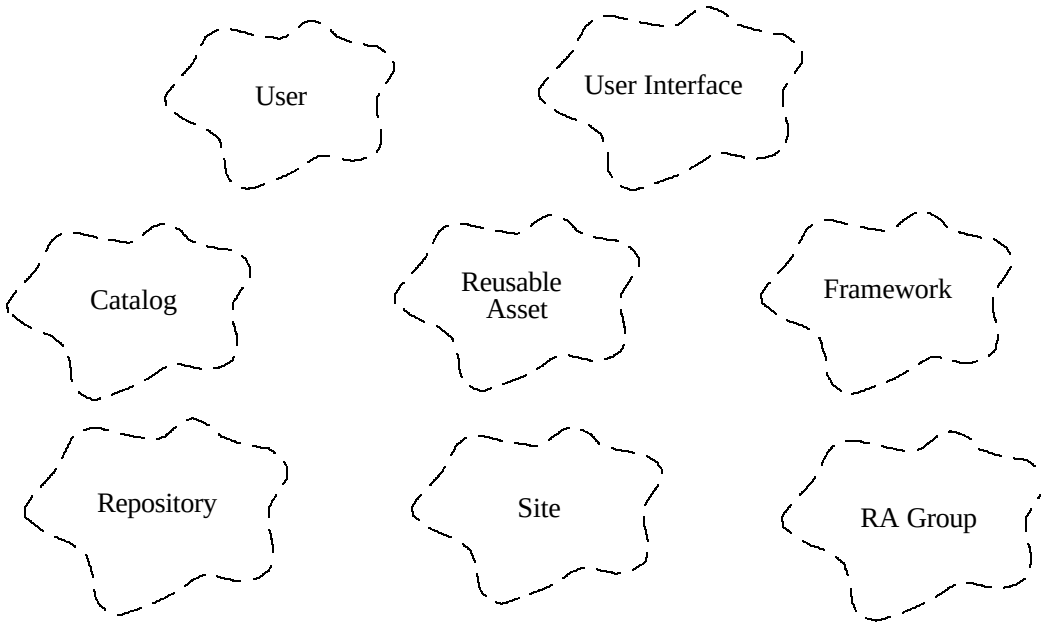

Figure 4-1. Key Classes in the DSRS

4.3.2 DSRS Class Diagram with Relationships. In the DSRS, there are two types of relationships. The first of these is association, which denotes some semantic dependency among classes. The second is aggregation, which denotes a "part of" relationship. Refer to Appendix B for definitions of relationships among classes. Figure 4-2 illustrates DSRS class diagrams with relationships. This class diagram is a preliminary diagram produced during the analysis stage. During the design stage, the class diagram was modified to reflect the understanding of the system at that time.

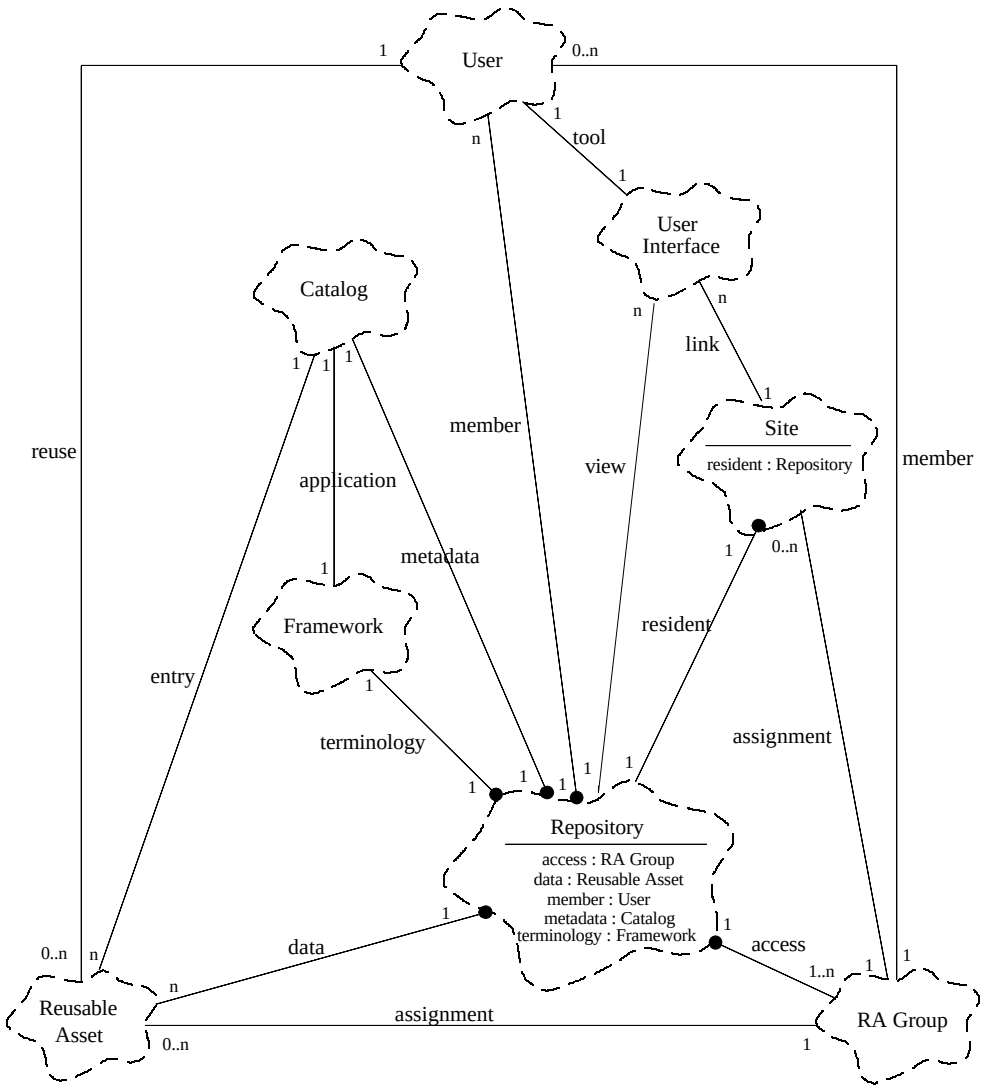

Figure 4-2. Class Diagram with Relationships

4.3.3 <u>DSRS Class Diagram with Operations</u>. Operation denotes services provided by the class. It is a functions or procedure that performs a task. Each operation is a member of a class. In Figure 4-3, operations that expose important characteristics of a class are indicated on the class diagram by their addition to the class compartments. A class compartment is a type of adornment that is displayed inside the class icon. Section 4.3.5 lists the complete set of operations for each classes.

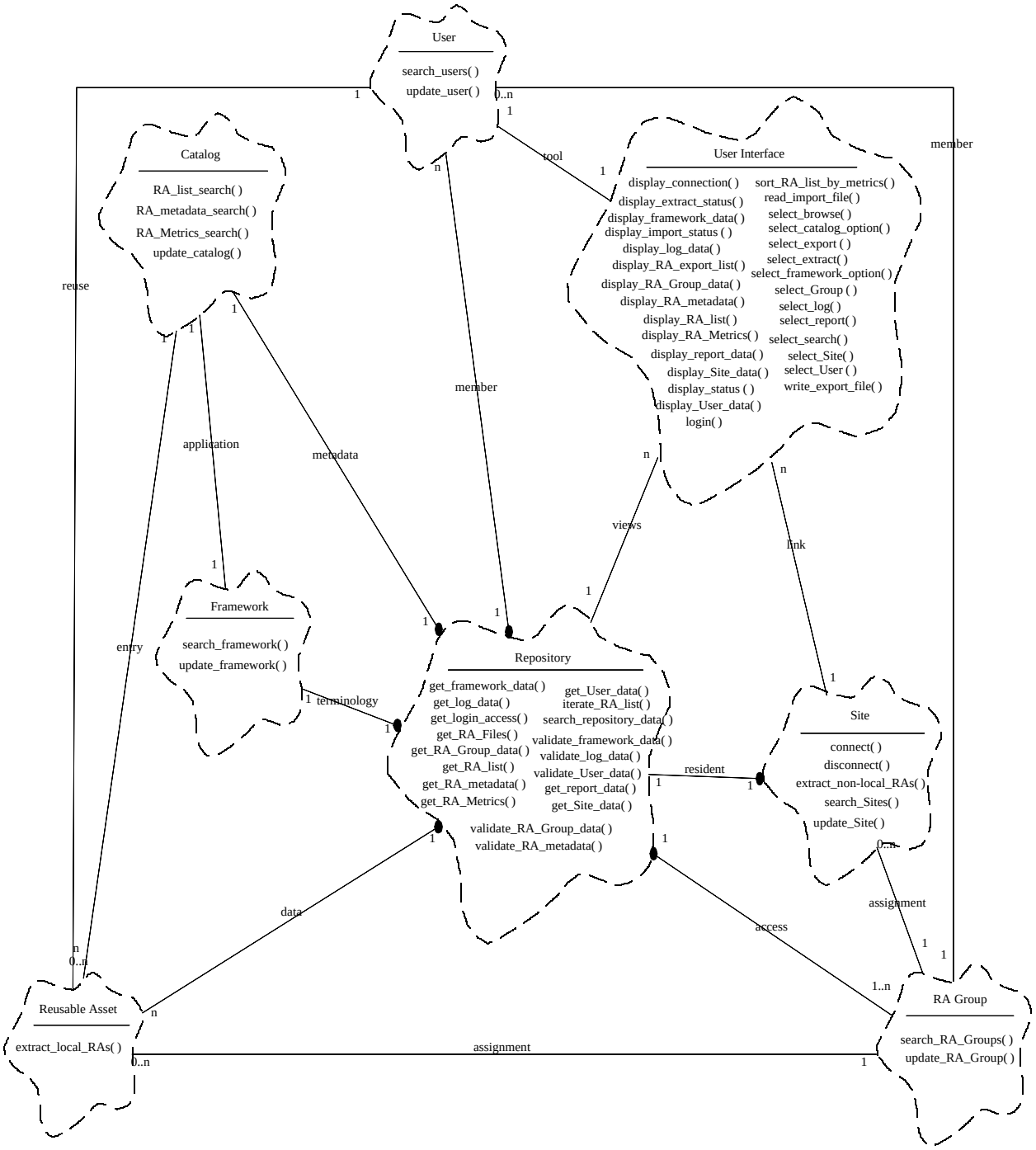

Figure 4-3. DSRS Class Diagram with Operations

4.3.4 DSRS Class Diagram with Attributes. Attributes are properties that describe a class. Figure 4-4 shows the DSRS class diagram with the attributes that represent key properties of the class. The complete attributes listing can be found in Section 4.3.5.

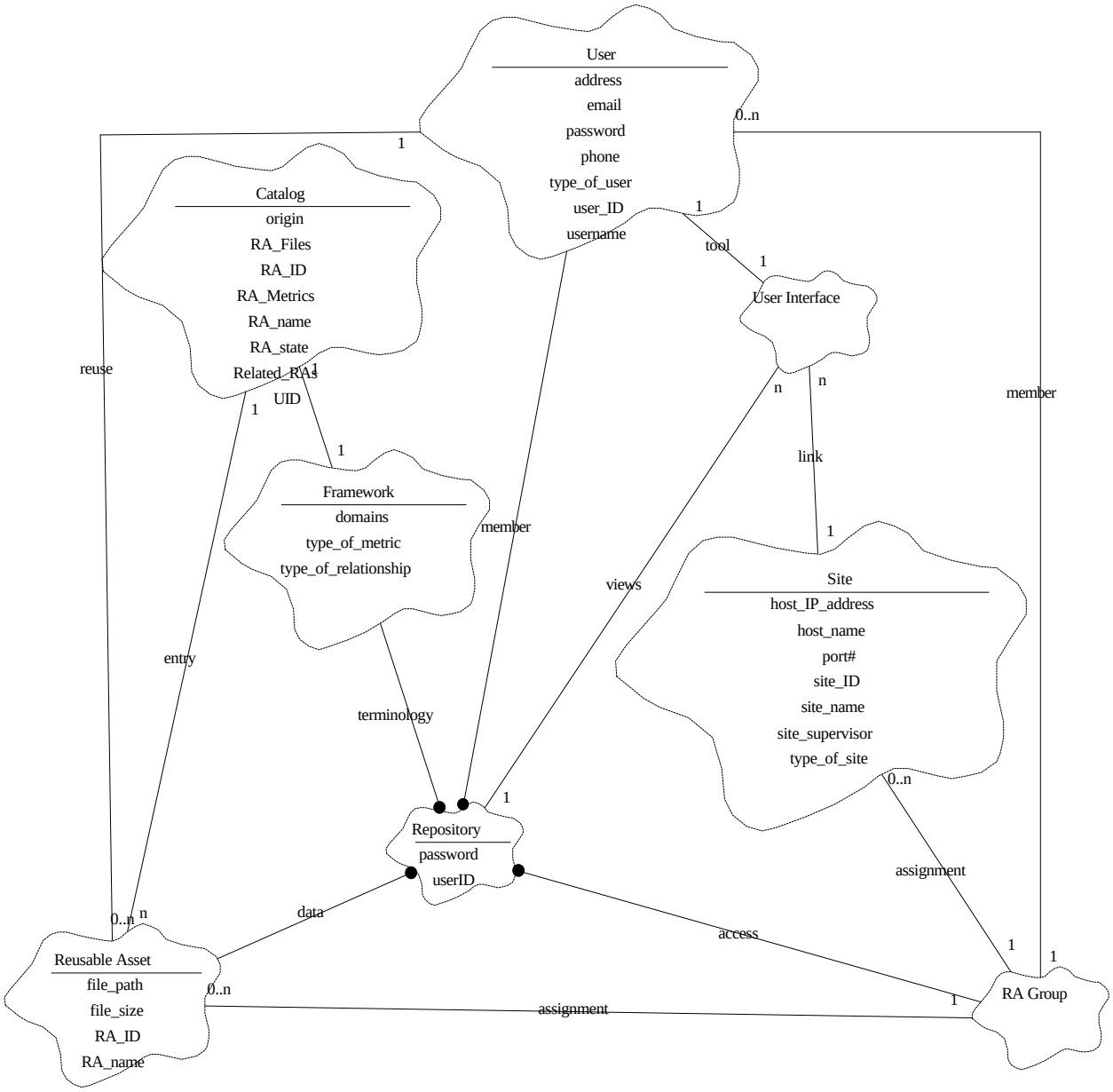

Figure 4-4. DSRS Class Diagram with Attributes

4.3.5 DSRS Class Specifications.

4.3.5.1 Class name: Catalog

a. Description: A catalog is a systematized list of Reusable Assets with descriptive information (metadata). Note: there are three types of catalog: repository catalog, local database, and WWW. WWW is a subset of catalog (operations: browse, search)

b. Operations:

(1) RA_list_search ( )
(2) RA_metadata_search ( )
(3) RA_Metrics_search ( )
(4) update_catalog ( )

c. Attributes:

(1) allow_viewing
(2) file_format
(3) file_name
(4) file_size
(5) file_title
(6) file_type
(7) file_version
(8) number_of_hits
(9) RA_ID
(10) RA_Metrics
(11) RA_name
(12) RA_state
(13) RA_version
(14) Related_RAs
(15) total_size
(16) UID

4.3.5.2 Class name: Framework

a. Description: A framework is the skeletal support (site configurable set of possible properties of a reusable asset) used as a basis for constructing a catalog.

b. Associations:

(1) Catalog <application>

c. Operations:

(1) search_framework ( )
(2) update_framework ( )

d. Attributes:
(1) domain_definition
(2) domain_ID
(3) domain_name
(4) generate_trace
(5) relation inverse
(6) type_of_metric
(7) type_of relationship

4.3.5.3 Class name: RA Group

a. Description: An RA group is the logical association of users, sites, and reusable assets.

b. Associations:
(1) User <member>
(2) Reusable Asset <assignment>

c. Operations:
(1) search_RA_Groups ( )
(2) update_RA_Group ( )

d. Attributes:
(1) temp_assigned_RAs
(2) assigned_sites
(3) assigned_users
(4) group_description
(5) group_ID

4.3.5.4 Class name: Repository

a. Description: A repository is a logical composition of users, reusable assets, and a framework to catalog descriptive information to support and maintain authorized use.

b. Associations:
(1) User Interface <views>

c. Has-A Relationships:
(1) RA Group <access>
(2) Reusable Asset <data>
(3) User <member>
(4) Catalog <metadata>
(5) Framework <terminology>

d. Operations:
(1) get_framework_data ( )
(2) get_log_data ( )

(3) get_login_access ( )
(4) get_RA_Files ( )
(5) get_RA_Group_data ( )
(6) get_RA_list ( )
(7) get_RA_metadata ( )
(8) get_RA_Metrics ( )
(9) get_report_data ( )
(10) get_Site_data ( )
(11) get_User_data ( )
(12) iterate_RA_list ( )
(13) search_repository_data ( )
(14) validate_framework_data ( )
(15) validate_log_data ( )
(16) validate_RA_Group_data ( )
(17) validate_RA_metadata ( )
(18) validate_Site_data ( )
(19) validate_User_data ( )

e. Attributes:
(1) browse_type
(2) login_access
(3) login_threshold
(4) RA_search_criteria
(5) RA_search_type

4.3.5.5 Class name: Reusable Asset

a. Description: A reusable asset is a resource in the repository that has potential value to the user.

b. Associations:
(1) Catalog <entry>

c. Operations:
(1) extract_local_RAs ( )

d. Attributes:
(1) allow_viewing
(2) author
(3) date_submitted
(4) date_updated
(5) file_format
(6) file_name
(7) file_size
(8) file_title
(9) file_type

(10) file_version
(11) POC
(12) RA_ID
(13) RA_Metrics
(14) RA_name
(15) RA_state
(16) RA_version
(17) Related_RAs
(18) total_size
(19) UID

4.3.5.6 Class name: Site

a. Description: A site is a distributed reusable asset (RA) repository that can accomodate multiple server sites and multiple users on each server.

b. Associations:
(1) RA Group <assignment>

c. Has-A Relationships:
(1) Repository <resident>

d. Operations:
(1) connect ( )
(2) disconnect ( )
(3) extract_non-local_RAs ( )
(4) search_Sites ( )
(5) update_Site ( )

e. Attributes:
(1) host_IP_address
(2) host_name
(3) port_num
(4) site_ID
(5) site_name
(6) site_supervisor
(7) suggestion_e_mail
(8) type_of_site

4.3.5.7 Class name: User

a. Description: A user is a member of the repository whose membership in a Reusable Asset (RA) Group provides access.

b. Associations:
(1) Reusable Asset <reuse>

(2) User Interface <tool>

c. Operations:

(1) search_users ( )
(2) update_user ( )

d. Attributes:

(1) city
(2) comments
(3) country
(4) e_mail
(5) expiration_date
(6) last_login_date
(7) login_failures
(8) number_of_logins
(9) organization
(10) password
(11) password_duration
(12) phone
(13) state
(14) street_address
(15) type_of_user
(16) user_ID
(17) user_name
(18) zip_code

4.3.5.8 Class name: User Interface

a. Description: A user interface is a graphical user interface (GUI) tool that provides a communication link with a site

b. Associations:

(1) Site <link>

c. Operations:

(1) display_connection ( )
(2) display_extract_status ( )
(3) display_framework_data ( )
(4) display_import_status ( )
(5) display_log_data ( )
(6) display_RA_export_list ( )
(7) display_RA_Group_data ( )
(8) display_RA_list ( )
(9) display_RA_metadata ( )
(10) display_RA_Metrics ( )
(11) display_report_data ( )

(12) display_site_data ( )
(13) display_status ( )
(14) display_user_data ( )
(15) login ( )
(16) read_import_file ( )
(17) select_browse ( )
(18) select_catalog_option ( )
(19) select_export ( )
(20) select_extract ( )
(21) select_framework_option ( )
(22) select_Group ( )
(23) select_log ( )
(24) select_report ( )
(25) select_search ( )
(26) select_Site ( )
(27) select_User ( )
(28) sort_RA_list_by_metrics ( )
(29) write_export_file ( )

d. Attributes:

(1) active_screen
(2) connectivity
(3) current_domain
(4) metric_sort_type
(5) selected_metric
(6) selected_RA_list
(7) session_RA_list

4.3.6 DSRS Object-Scenario Diagrams. Object-Scenario Diagrams describes how objects collaborate to realize a use case. They trace the execution of a scenario. The arcs of the object-scenario diagrams represent messages passed between objects. They are labeled with the name of operations. Sequence numbers are added to show the relative ordering of the messages.

As described in the system function statement section, there are four use cases pertaining to the DSRS User Tool, eight use cases pertaining to the DSRS Librarian Tool and four use cases for the DSRS Server. This section will show a scenario for each of the use case.

4.3.6.1 DSRS User Object-Scenario Diagrams.

4.3.6.1.1 Type of Catalog Search Object Diagram. The object-scenario diagram in Figure 4-5 describes the process of selecting a search on the catalog. A user can select a search by List RA IDs, List RA Names, Find Keywords, or Catalog Indexes on the World Wide Web (WWW) to build a list of candidate RAs.

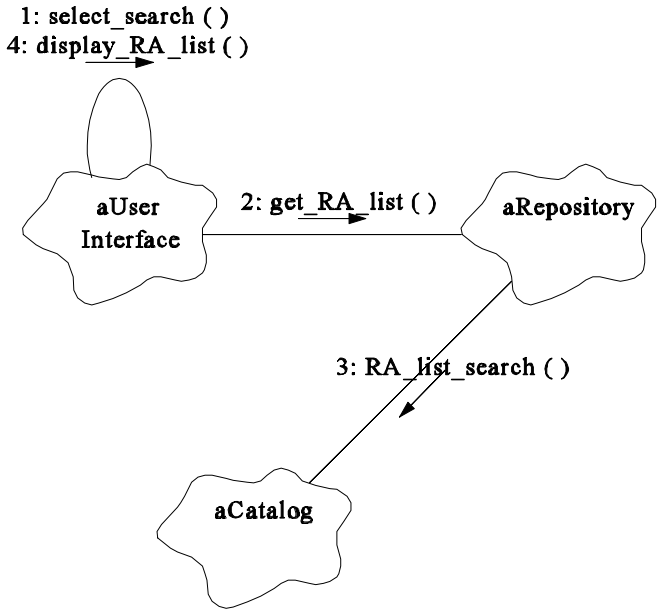

Figure 4-5. Type of Catalog Search Object Diagram

4.3.6.1.2 Type of Browse List Object Diagram. The object-scenario diagram in Figure 4-6 describes the process of selecting an RA entry from the list of candidates to browse the catalog metadata by File List, Metrics, Related RA List, or view the catalog page on the World Wide Web (WWW).

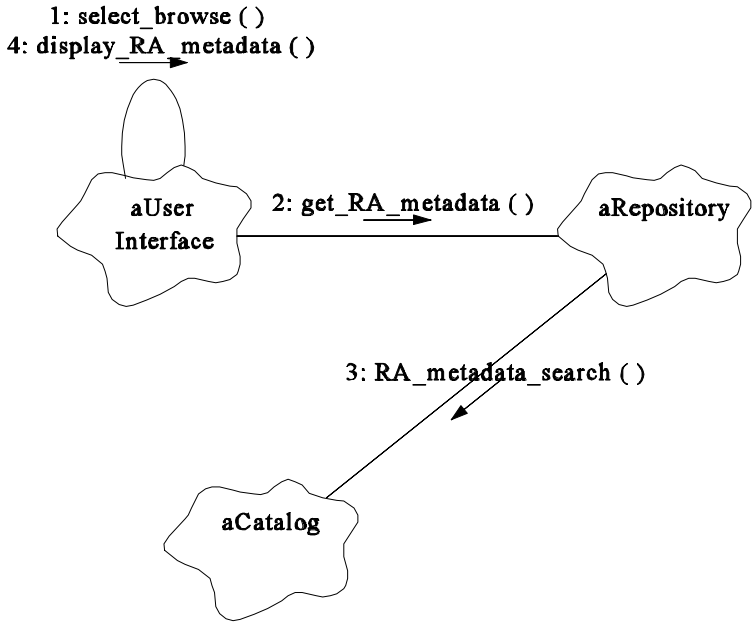

Figure 4-6. Type of Browse List Object Diagram

4.3.6.1.3 Type of Metrics Analyze Object Diagram. The object-scenario diagram in Figure 4-7 describes the process of selecting RA entries from the list of candidates to graphically analyze metrics.

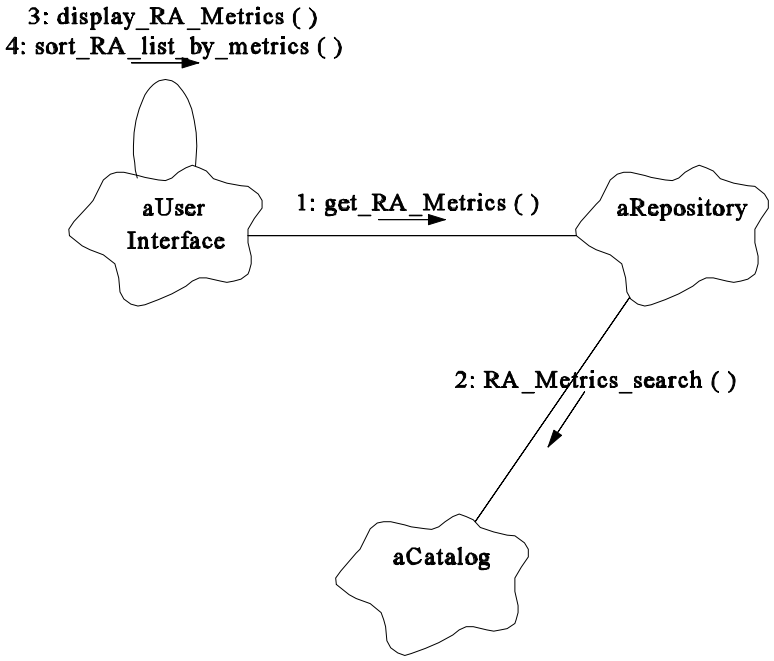

Figure 4-7. Type of Metrics Analyze Object Diagram

4.3.6.1.4 Reuse Object Diagram. The object-scenario diagram in Figure 4-8 describes the process of selecting RA entries from the list of candidates to extract RA Files.

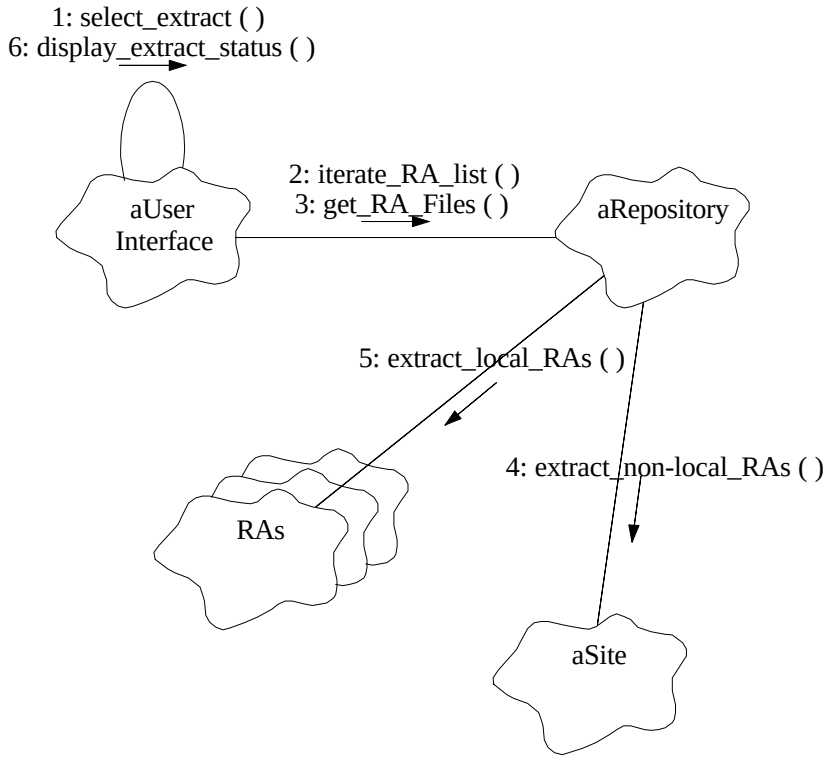

Figure 4-8. Reuse Object Diagram

4.3.6.2 DSRS Librarian Object-Scenario Diagrams.

4.3.6.2.1 Type of Framework Object Diagram. The object-scenario diagram in Figure 4-9 describes the process of maintaining repository framework. Repository framework includes: types of relationship and types of metrics.

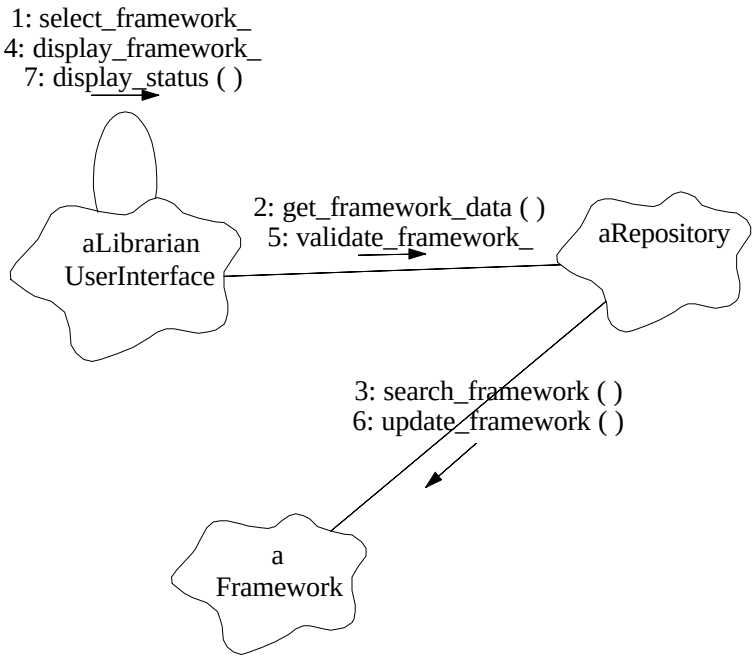

Figure 4-9. Type of Framework Object Diagram

4.3.6.2.2 Import Object Diagram. The object-scenario diagram in Figure 4-10 describes the process of writing an import file.

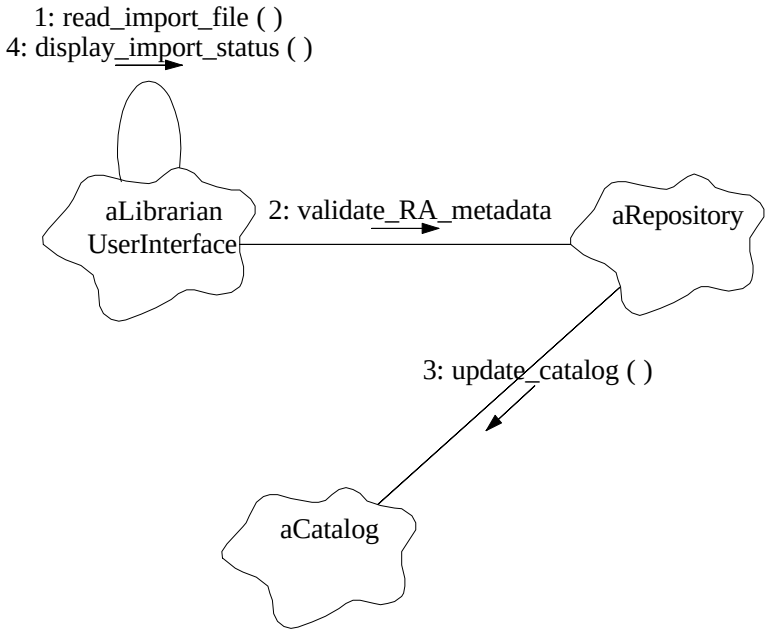

Figure 4-10. Import Object Diagram

4.3.6.2.3 Export Object Diagram. The object-scenario diagram in Figure 4-11 describes the process of creating an export file containing RA metadata that can be imported into another DSRS catalog.

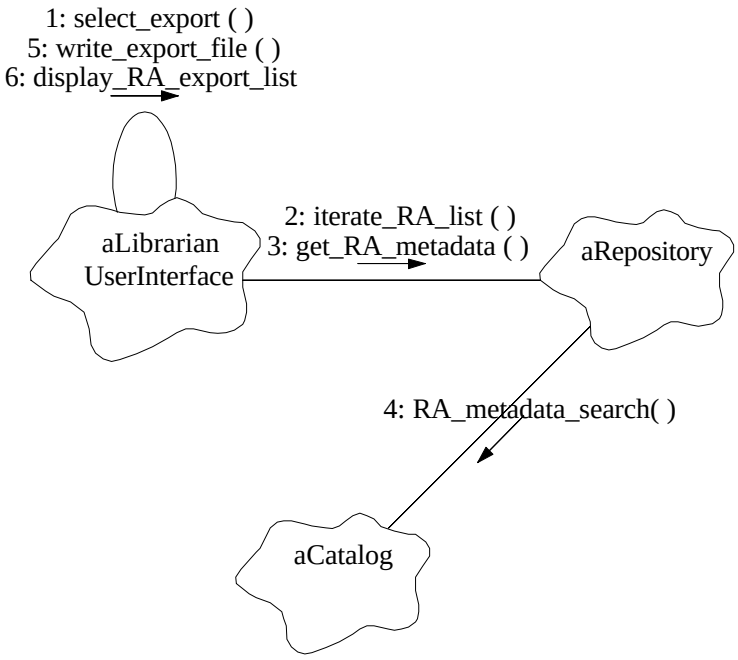

Figure 4-11. Export Object Diagram

4.3.6.2.4 Type of RA Group Object Diagram. The object-scenario diagram in Figure 4-12 describes the process of maintaining group information and group assignments among sites, users, and RAs.

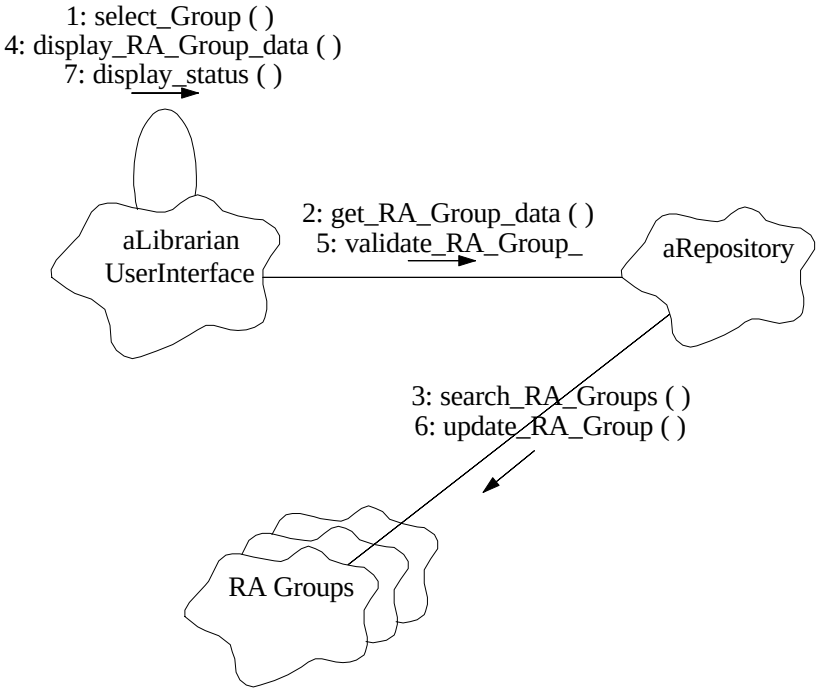

Figure 4-12. Type of RA Group Object Diagram

4.3.6.2.5 Type of Users Object Diagram. The object-scenario diagram in Figure 4-13 describes the process of maintaining user information.

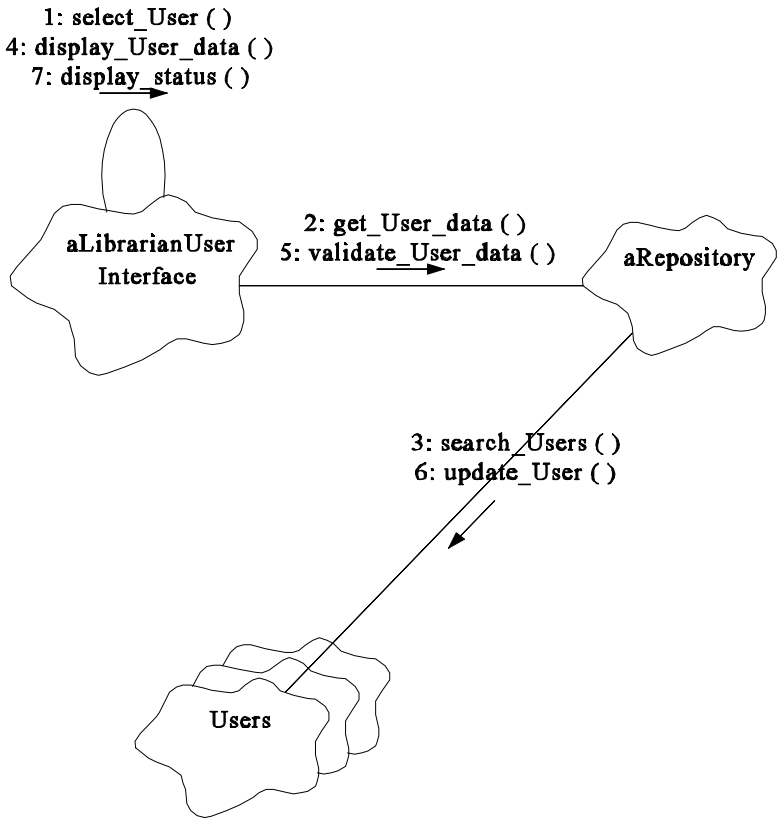

Figure 4-13. Type of Users Object Diagram

4.3.6.2.6 Maintain Logs Object Diagram. The object-scenario diagram in Figure 4-14 describes the process of maintaining the entries in the use log.

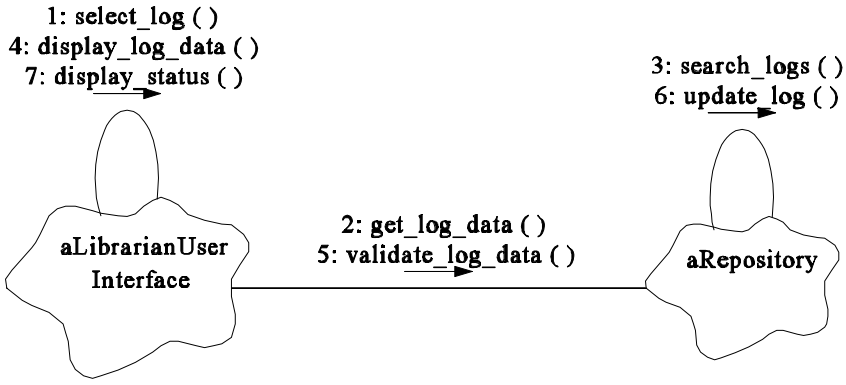

Figure 4-14. Maintain Logs Object Diagram

4.3.6.2.7 Type of Sites Object Diagram. The object-scenario diagram in Figure 4-15 describes the process of maintaining site information for the local site, remote DSRS sites, and foreign sites.

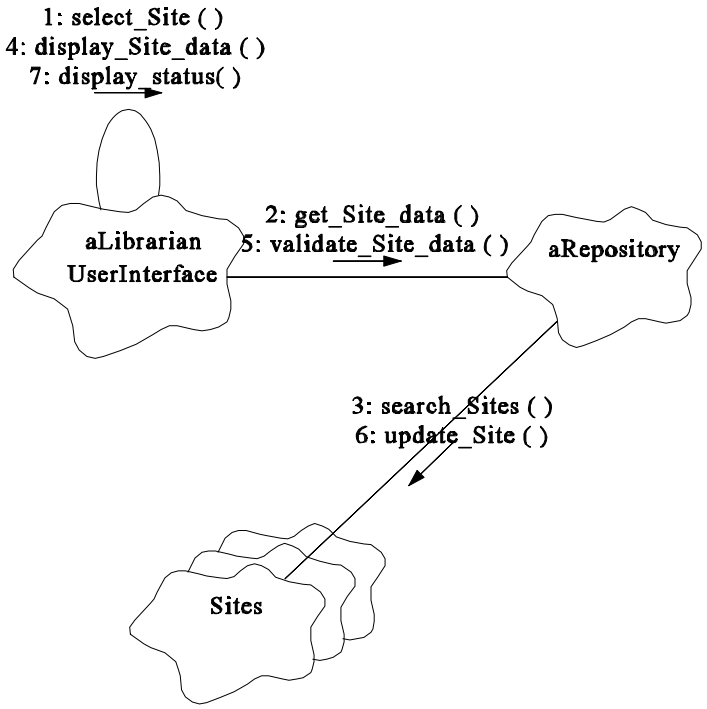

Figure 4-15. Type of Sites Object Diagram

4.3.6.2.8 Generate Reports Object Diagram. The object-scenario diagram in Figure 4-16 describes the process of providing predefined and customized report capability on data in the repository.

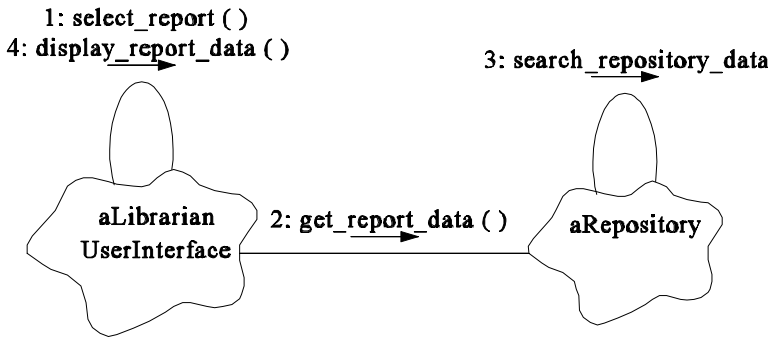

Figure 4-16. Generate Reports Object Diagram

4.3.6.3 DSRS Server Object-Scenario Diagrams.

4.3.6.3.1 Type of Connectivity Object Diagram. The object-scenario diagram in Figure 4-17 describes the process of providing connectivity capability to a remote DSRS server site to gain access to the information stored in the catalog.

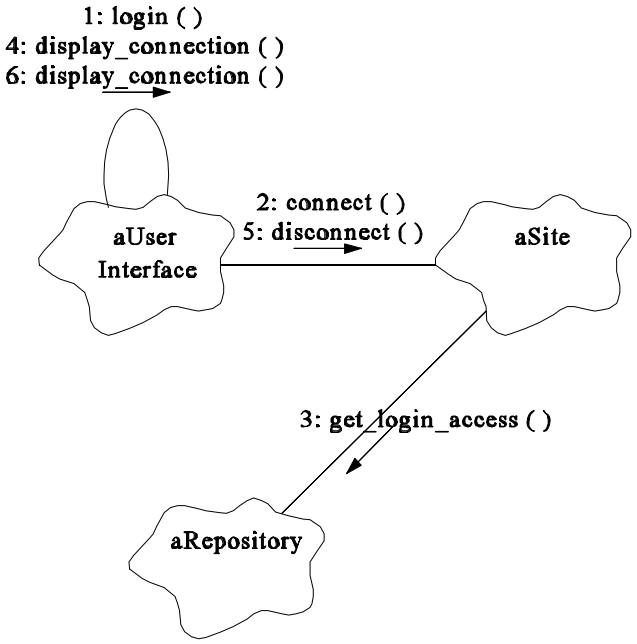

Figure 4-17. Type of Connectivity Object Diagram

4.3.6.3.2 Type of Metadata Object Diagram. The object-scenario diagram in Figure 4-18 describes the process of processing catalog requests for RA metadata. RA metadata includes: RA Metrics, list of related RAs, list of RA Files, and other RA Information in the catalog.

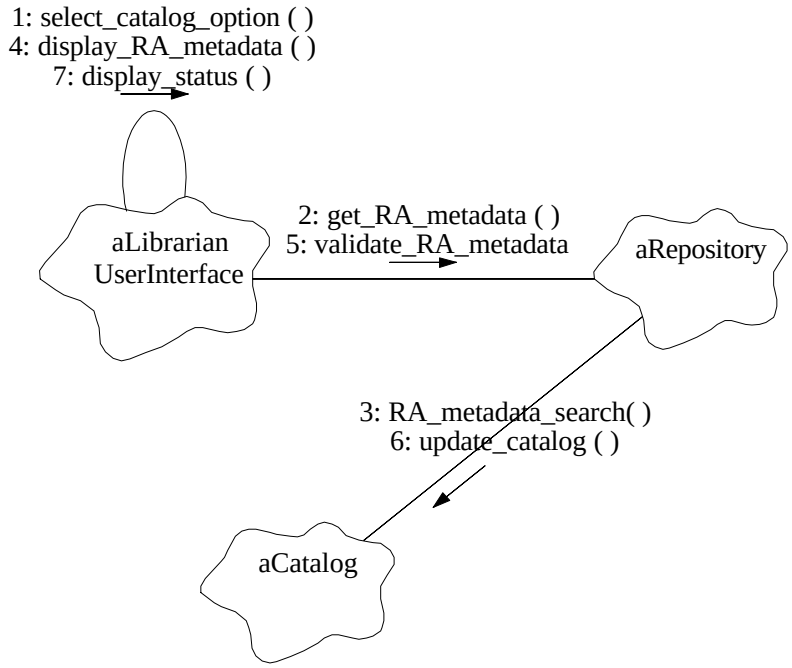

Figure 4-18. Type of Metadata Object Diagram

4.4 SYSTEM DESIGN. System design is the process of expanding what was learned during domain analysis into a working implementation. Designing a system entails a series of decision on what is the most cost-effective implementation that will carry out the system charter and lead to reuse among many systems. The products of this process are: architectural descriptions, class category diagrams, and design class diagrams.

During the analysis stage of the DSRS development process, a preliminary top-level class diagram was produced (See Figure 4-2). It incorporated the classes for the *User* and *Librarian* tools. At the design stage, this preliminary top-level diagram was amended. There are separate top-level class diagrams for the DSRS User for MS-Windows, DSRS User for X/Motif, and DSRS Librarian for MS-Windows (See Figure 2-6, Figure 2-7, and Figure 2-8).

4.4.1 Architectural Description. Architectural descriptions capture major design decisions such as choice of database managers, operation systems, language, and so on. The architecture of DSRS is organized in layers. Layers are divided into loosely coupled partitions, each providing one kind of service. In the Booch method, these partitions are called class categories. Class categories are clusters of highly related classes that are themselves cohesive, but are loosely coupled related to other such clusters.

4.4.1.1 DSRS User Architectural Description.

Server:

Languages: C, Pro*C, Yacc, Lex

Application:

(1) *MS-Windows Version:*
- GUI Interface: Visual Basic
- Language: Basic

(2) *X/Motif Version:*
- GUI Interface: XVT Development Solution for C++
- Language: C++

WWW:

Web Browser (e.g., Mosaic, Netscape)

Local Database:

(1) *MS-Windows Version:*
- Access

(2) *X/Motif Version:*
- MSQL

Help:

(1) *MS-Windows Version:*
- RoboHELP
- Visual Basic

(2) *X/Motif Version:*
- XVT Help Engine

- XVT Development Solution for C++

4.4.1.2 DSRS Librarian Architectural Description.

Server:

- Languages: C, Pro*C, Yacc, Lex

Application:

- GUI Interface: Visual Basic
- Language: Basic

Report:

- ReportSmith

Help:

- RoboHELP
- Visual Basic

4.4.2 DSRS Class Category Diagrams. Figure 4-19 and Figure 4-20 describe the class category diagrams for the DSRS.

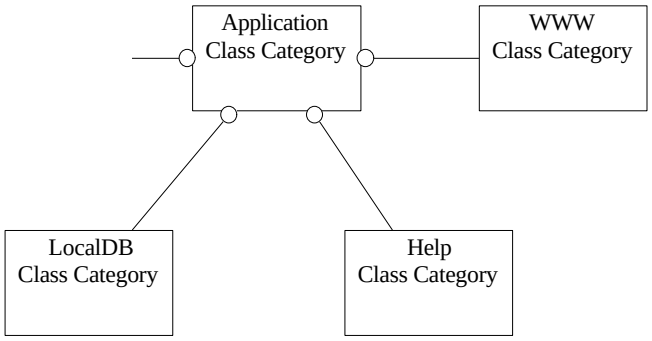

Figure 4-19. DSRS User Class Category Diagram

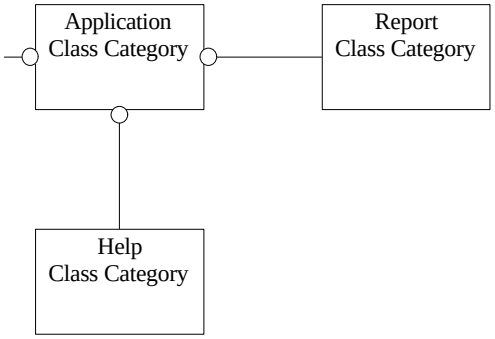

Figure 4-20. DSRS Librarian Class Category Diagram

4.4.3 AMENDED DSRS CLASS DIAGRAMS. During the third step of the Booch method, *System Design*, the preliminary top-level class Diagram described in Figure 4-2 is broken down into the top-level design class diagrams for the *User* tools (MS-Windows and X/Motif Versions), and the top-level class diagram for the *Librarian* tool (MS-Windows Version). Refer to Figure 2-6, Figure 2-7, and Figure 2-8.